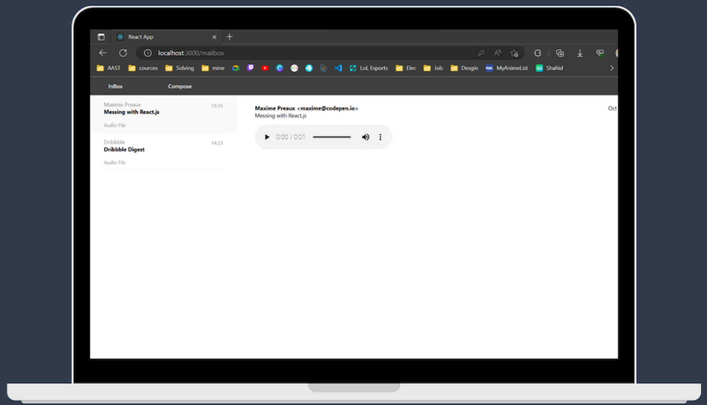
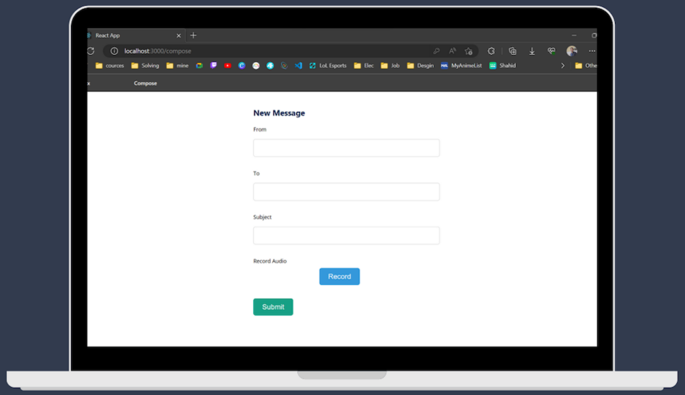
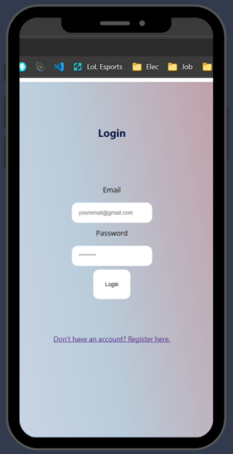
Voice Mail System

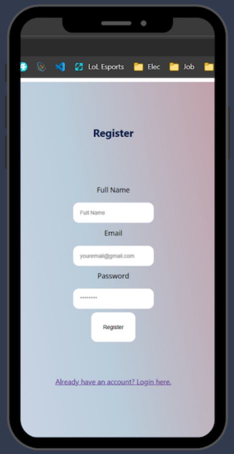
Client Side

On the client side, React.js, a popular JavaScript library, was chosen for building a dynamic and interactive user interface. React-Mic, a React component, was integrated to facilitate audio recording capabilities within the application. Socket.io was also utilized on the client side to establish a seamless connection with the server, enabling real-time message updates and notifications.









Project Description

The project involves the implementation of a client-server voice mail application that enables users to send voice messages to each other. This application will provide a convenient and efficient way for users to communicate using recorded voice instead of traditional text-based messages. Users will be able to record voice mails, send them to other users, and receive voice mails from others within the application.

Project Architecture

The client-server architecture allows for seamless communication between users by leveraging a server to handle message transmission and storage. The server-side logic will be responsible for receiving incoming voice mails, storing them securely, and facilitating their delivery to the intended recipients. On the client side, users will have access to an intuitive interface that allows them to record, send, and receive voice mails, as well as manage their mailbox.

Project Tech Stack

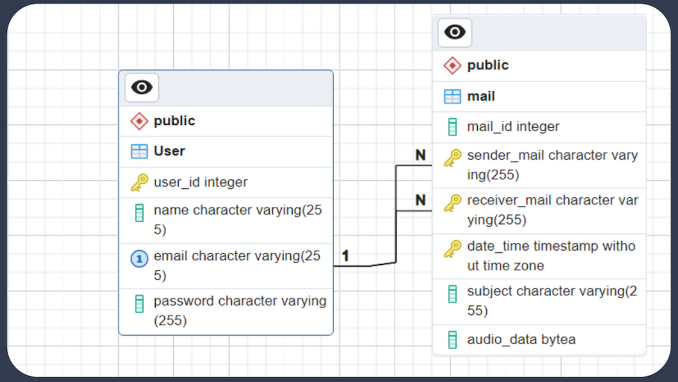
Done By:

Name	Reg#
Ahmed Hosam	19103378
Ahmed Ramadan	19108101
Abdelrahman Mohamed Mostafa	19108100
Mahmoud Awad Saad	19101863
Farah Ahmed Anany	19103787
Khloud Maged	19102967

Server Side

On the server side, Node.js served as the foundation, allowing for scalable and asynchronous event-driven programming. The Express framework was utilized to simplify the development of the server-side logic, enabling efficient routing and handling of HTTP requests.
Socket.io was employed to establish real-time bidirectional communication between the server and clients, facilitating instant message transmission and updates. PostgreSQL, a powerful relational database management system, was utilized for storing and managing voice mail data securely and efficiently.
To ensure secure user authentication, JSON Web Tokens (JWT) were implemented, providing a mechanism to verify and authorize users accessing the application.

Database ERD


public
User
user_id integer
name character varying(255)
email character varying(255)
password character varying(255)
public
mail
mail_id integer
sender_mail character varying(255)
receiver_mail character varying(255)
date_time timestamp without time zone
subject character varying(255)
audio_data bytea
N
N
1

Socket Events

- userConnected: emitting the "userConnected" event to the client connected to the socket, providing information about the user's ID.
- sendMail : performs necessary logic for sending an email, notifies the recipient client about the received mail, and sends a response back to the client that sent the original "sendMail" event.
- mailSent
- getMails : retrieves the emails associated with the user, and sends them back to the client through the "mailReceived" event.
- mailReceived